Generation of Transfer Functions with Stochastic Search Techniques

Generation of Transfer Functions with Stochastic Search Techniques

Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister

Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794–4400

Abstract

This paper presents a novel approach to assist the user in exploring appropriate transfer functions for the visualization of volumetric datasets. The search for a transfer function is treated as a parameter optimization problem and addressed with stochastic search techniques. Starting from an initial population of (random or pre-defined) transfer functions, the evolution of the stochastic algorithms is controlled by either direct user selection of intermediate images or automatic fitness evaluation using user-specified objective functions. This approach essentially shields the user from the complex and tedious "trial and error" approach, and demonstrates effective and convenient generation of transfer functions.

1 Introduction

Direct volume rendering is a key technology for the visualization of large sampled, simulated, or synthesized 3D datasets from scientific, engineering, and medical applications. However, an important problem that inhibits its widespread use is the complex interrelationship of rendering parameters combined with the lack of interactivity. For example, the user often has to find the appropriate parameter settings for the rendering of a dataset without immediate visual feedback.

Of particular importance to the outcome of direct volume rendering are transfer functions. A transfer function assigns values for optical properties, such as color and opacity, to original values of the dataset being visualized. In certain applications, the appropriate choice of transfer function depends to a large extent on the data itself. For example, in routine medical visualizations of CT data it is often possible to use pre-defined transfer functions to highlight certain tissue types, such as bone or muscle. However, when using visualization for data exploration, the image depends to a large extent on the subjective goals of the user. For example, molecular scientists may want to visualize a molecule in vastly different ways, depending on their needs. In these cases, it is imperative to experiment with different transfer functions before a satisfactory visualization is achieved. This exploration of the parameter space can also lead to unexpected discoveries about the data.

By far the most commonly used method for data exploration is "trial and error." Transfer functions are repeatedly modified and each newly-rendered image is judged with the hope of eventually finding a good visualization, which highlights some properties of the volume data in an informative way. In this paper, we introduce a fundamentally new method for the generation of transfer functions. In general, we treat the search for the appropriate transfer function as a parameter optimization problem which can be approached with stochastic search techniques. Figure 1 shows the two new methods that will be explored in this paper.

Figure 1a shows the visualization process using semi-automatic generation of transfer functions. Stochastic algorithms are used to generate a first set (or population) of transfer functions, which leads to a first population of images. Based on user evaluation of the images, the search process is repeated until a satisfactory visualization has been achieved. We have incorporated this approach into an intuitive and easy-to-use user interface. In Section 5, we show results from our experiments. They indicate that an optimal visualization is achievable even though the only user interaction is image evaluation.

In an alternative approach, shown in Figure 1b, the visualization process allows the user to express the objective goals for the desired images. Examples of objective functions include entropy, histogram variance,

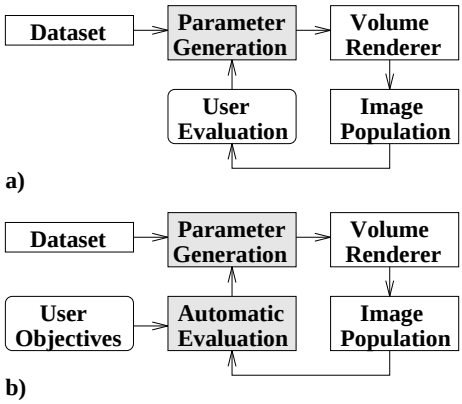

Figure 1: a) Automatic parameter generation with user evaluation of the resulting images. b) Automatic parameter generation and automatic evaluation of images based on user-defined objective functions.

or edge energy of the resulting images. Both of these methods allow the user to evaluate the visualization, either subjectively or objectively, in terms of the final images, and not in terms of the shape or physical interpretation of the underlying transfer functions.

The next section defines transfer functions in the context of direct volume rendering, and Section 3 introduces the key concepts of stochastic search techniques. We show how to use stochastic algorithms to generate a set of transfer functions. Section 4 presents our implementation, which incorporates the genetic evolution methods into an easy to use and intuitive user interface. Finally, Section 5 discusses the results by applying several well-known stochastic search algorithms to the problem of transfer function generation.

2 Transfer Functions

Volume rendering techniques typically rely on the low-albedo approximation to how the volume data generates, scatters, or occludes light [4, 9]. Effects of the light interaction at each dataset location are integrated continuously along viewing rays according to the following equation:

$$I(a,b) = \int_a^b s(x) e^{\int_a^x \alpha(t)\,dt}\,dx, \qquad (1)$$

where

$I(a,b)$ is the intensity from a ray passing through the dataset between points a and b;

$s(x)$ is the source term, giving the light added per unit length along the ray, including self-emission and reflected light; and

$\alpha(t)$ is the absorption coefficient, corresponding to the attenuation of light per unit length along the ray due to scattering or extinction.

The mapping which assigns a value for optical properties like α or s to each value of the scalar f being visualized is called a *transfer function*. The transfer function for α is called *opacity transfer function*, typically a continuously varying function $\alpha(f)$ of the scalar f. Often it is also useful to include the surface normal, approximated by the direction of the gradient ∇f, as an additional parameter of the opacity transfer function $\alpha(f, \nabla f)$. This approach has been widely used in the visualization of bone or other tissues in medical datasets or for the iso-surface visualization of electron density maps [11].

The source term s can also be specified as a transfer function $s(f)$ of the scalar f. The simplest source term is direction independent, representing the glow of a hot gas [12]. It may have red, green, and blue components, with their associated *color transfer functions* $s_{\text{red}}(f)$, $s_{\text{green}}(f)$, and $s_{\text{blue}}(f)$. More sophisticated models include shading effects, where the source term at position x is calculated as $s(x) = \rho(x, \omega_r, \omega_i) i(x)$. $i(x)$ is the incoming illumination reaching x, and $\rho(x, \omega_r, \omega_i)$ is the bidirectional spectral reflectivity, as a function of the direction ω_r of the reflected light, the direction ω_i of the incoming light, and other properties that vary with x, such as the surface normal [12]. Common approximations for ρ include the Phong [14], Blinn [3], or Cook and Torrance [5] shading models. All of these models may contain one or more rather complex transfer functions that can not be analytically expressed.

Despite these physical interpretations of transfer functions, it is often difficult for the user to select them in a meaningful way. For example, when no prior knowledge about the dataset is available, different transfer functions may lead to different discoveries of interesting properties of the data. In addition, the complex interaction of opacity and emission makes it very hard, if not impossible, to predict the effect of changing more than one of them simultaneously. This work is aimed at overcoming this problem by providing additional tools for data exploration that do not require any understanding of the underlying physical processes.

3 Stochastic Search Techniques

We view the search for the appropriate transfer function as a parameter optimization problem where meth-

ods developed for global optimization can be used. The domain space for transfer function selection is very large, with the number of possibilities growing exponentially with the number of transfer functions. Because of the long volume rendering time for each image, a fast convergence to the global optimal solution becomes essential. This excludes an exhaustive search among all possible solutions. It also prohibits the use of enumeration techniques such as dynamic programming or random search techniques such as Monte-Carlo methods. Furthermore, we would like to approach the optimum by an adaptive, stepwise strategy. Partially satisfying, intermediate results should be passed on to the next iteration with the intention of positively influencing the final image.

Stochastic search techniques are particularly suitable for this purpose. Although there is no formal guarantee of convergence, stochastic search techniques have a high probability of locating the global solution optimally in a multi-modal search landscape, where several locally optimal solutions exist. They have been successfully used in many fields, including computer graphics [15]. The implementation of stochastic algorithms is remarkably easy. The basic structure of the methods we investigate in this paper is shown in Figure 2.

First, we are looking for an *encoding* of the optimization problem solutions (**1**), which can be thought of as a unique mapping of structures $x(i)$ onto solutions. In the simplest case, each structure is a binary string. In the following subsection we present conditions for the appropriate encoding of transfer functions. A *population* $P(t)$ at time t is an S-dimensional vector of structures

$$P(t) = \langle x_1(t), x_2(t), \ldots, x_S(t) \rangle, \qquad (2)$$

where S is the *population size*. The *initial population* $P(0)$ (**2**) is usually chosen at random to represent a wide variety of encodings. Alternatively, heuristically chosen initial transfer functions may be used to an advantage. For the generation of solutions from encodings it is customary to use the biological terms *genotype* and *phenotype*. The genotype is the encoded representation of a possible problem solution into a structure $s_i(t)$, in this case an encoded representation of a transfer function. The encoded representations are used to produce phenotypes that give a user-defined meaning to the structures. In this case, the encoded transfer functions produce rendered images as phenotypes. The translation from genotype (encoded transfer function) to phenotype (image) is given by the direct volume rendering algorithm. An image is rendered for each of the transfer functions of the current population $P(t)$. Each image is then evaluated (**3** and **4c**), and a measure of success, or *fitness*, is assigned automatically or provided interactively by the user.

1. Design a schema to encode the solutions to the optimization problem.

2. Generate an initial population $P(0)$ of possible solutions.

3. Evaluate the initial population and assign a fitness value to each solution.

4. While no satisfactory solution is found:

- **4a.** Stochastically select an intermediate population $P'(t)$.
- **4b.** Generate new solution population $P(t)$ from $P'(t)$.
- **4c.** Evaluate the new solutions in $P(t)$.
- **4d.** Possibly load-balance the population of solutions.

Figure 2: Basic stochastic search algorithm.

Stochastic search algorithms mainly differ in how they select an intermediate population (**4a**), how the new solutions are generated (**4b**), and how a possible load-balancing of solutions is applied (**4d**). In this paper, we apply hill-climbing (HC), parallel hill-climbing (PH), parallel hill-climbing which attempts to balance the load by keeping good solutions over more than one iteration (PL), simulated annealing (SA) [10], and genetic algorithms (GA) [8]. We will now discuss in more detail how the basic stochastic search algorithm shown in Figure 2 can be applied to the generation of transfer functions.

3.1 Solution Encoding

In Section 2, we defined a transfer function as a mapping from the scalar data f and/or gradient to some optical property, such as color and opacity. In the following discussion we restrict ourselves to transfer functions that map a single scalar value to a single optical property. However, the algorithms can easily be extended to allow for multiple parameters, as we show in Section 5.

The domain and range of a transfer function can be normalized to be in the interval $[0, 1]$, so that we can

mathematically define a transfer function as:

$$f: D \rightarrow V, \qquad D \in [0,1],\ V \in [0,1]. \tag{3}$$

Although theoretically a transfer function can be of any form, optical models generally have some requirements for smoothness and continuity of valid transfer functions. Furthermore, more easily identified optimality conditions can be provided if f is a smooth function with continuous first and second derivatives. We call this constraint the *smoothness condition*.

For maximum generality we choose a vector representation of the continuous transfer functions. The N samples representing a transfer function f_i can be stored as a vector of N floating point numbers $x_i = [s_1, s_2, \ldots, s_N]$. This representation has to satisfy the following two reconstruction conditions. The original signal representing the function was band-limited and it has been sampled above the Nyquist frequency. If these conditions are met, we can perfectly reconstruct the continuous function f from N discrete samples. Stochastic changes are applied to individual values in this vector. To guarantee the smoothness condition, we apply a low-pass filter to any newly-generated vector.

The user may have previous knowledge about the data that allows transfer functions to take on certain values over specified domains but to exclude others. For example, the user may specify certain ranges of opacity values for different domains of the function that correspond to different tissue types. We call these pre-defined constraints *domain conditions*. They can be defined over different ranges of sample indices of the transfer function vector x_i, and are always satisfied in the whole optimization process.

To simultaneously generate transfer functions for more than one optical property we can reformulate the stochastic search as a single-objective problem either by forming a weighted combination of the different objectives or by replacing some of the objectives by constraints. Furthermore, instead of evaluating the image as a whole, one very often would like to combine aspects of one image, for example the transparency of an object, with aspects of another image, for example the color of the object. To provide this flexibility, we have developed an approach that we call *independent property selection*. The user has the option of interactively or automatically assigning different fitness values for each optical quality of an image, such as opacity or color.

3.2 Initial Population

The initial population of transfer functions $P(0)$ can be either randomly generated or provided by the user. Because certain transfer functions are commonly used as starting points, for example linear ramps, we provide a library of some typical initial functions. The user has the option of editing and changing these functions before proceeding with the stochastic process. Figure 3 shows some examples of initial transfer functions.

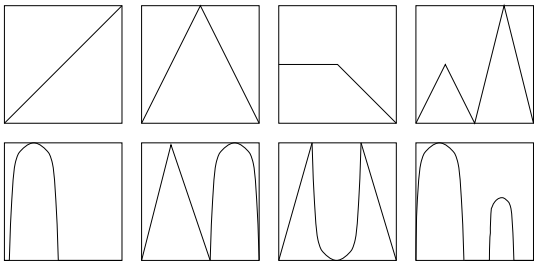

Figure 3: Examples of initial transfer functions.

3.3 Selection of the Intermediate Population

There are two methods for the selection of the intermediate population of solutions $P'(t)$. First, we can randomly pick individual solutions from the previous population with uniform probability distribution. This uniform random selection scheme is used for the hill-climbing (PH, PL) and the simulated annealing (SA) algorithms.

For the genetic algorithm we use *proportionate selection*, where a function with fitness value f_i is allocated $f_i/\bar{f}$ offsprings [8, 16]. $\bar{f}$ is the average fitness value of the population. If each selected function has a fitness value of $f_i = 1$, the average fitness becomes s/S', and the expected number of parents for each function is S'/s. A systematic conversion of this fractional number into the actual number of offsprings may result in methodical allocation biases. We use a stochastic *roulette wheel selection scheme* to generate the population $P'(t)$ of parents [2]. Each transfer function is allocated a slot of a roulette wheel. The size of the slot is proportionate to the expected number of parents S'/s. A random number $r \in [0, 2\pi]$ determines the slot of the next transfer function to be copied into the parent population $P'(t)$. This procedure is repeated until all parents are generated.

3.4 Generation of New Solutions

Every one of the N samples of each transfer function in $P'(t)$ is given a chance to undergo so-called *mutation* according to the mutation rate R_m. Mutation perturbs individual transfer functions to introduce stochastic variability [16]. In order to gradually decrease the search rate, we use a *dynamic mutation*

strategy, where the mutation rate R_m exponentially decreases with an increasing number of generations [16]. If an original floating point sample s_o is undergoing mutation, it is perturbed by a number p_m to the new sample value $s_n = s_o + p_m$. p_m is determined by a random perturbation factor $f_m \in [-1, 1]$ and a constant perturbation distance d_m, such that $p_m = f_m d_m$.

To satisfy the smoothness condition as discussed above in "Solution Encoding," we apply a Gaussian filter during the mutation operation, an approach we call *mutation splatting*. Figure 4 shows the general concept. The original sample s_o at position x_m is mu-

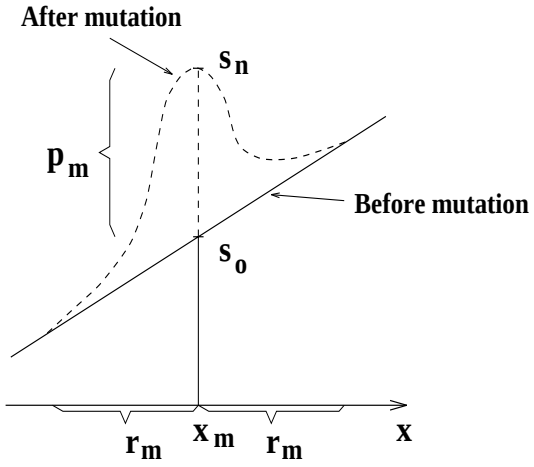

Figure 4: Mutation splatting at position x_m. s_o is the original sample; $s_n$is the new sample; p_m is the perturbation; r_m is the splatting range.

tated by a perturbation p_m to the new sample s_n. A random number r_m is generated and defines the range $[x_m - r_m, x_m + r_m]$. The Gaussian function $G(x)$ is applied inside this range, so that:

$$f(x) = f(x) + p_m G(x - x_m), \ (x_m - r_m) \leq x \leq (x_m + r_m).$$

Mutation splatting introduces a high variance to the new functions while preserving certain smoothness conditions. To compensate for the remaining spikes due to crossover, we apply a low-pass filter after reproduction.

In addition to mutation, which is used by all stochastic search techniques, genetic algorithms also use *recombination*, which merges information with the hope of producing more adapted solutions [16]. Recombination is achieved by the so-called *crossover* operator. The parents are randomly paired, and each pair is given a chance for crossover according to the crossover rate R_c. Among the different crossover mechanisms, we chose to implement *two-point crossover* [6]. The arrays of N samples encoding the transfer functions are treated as rings. Two crossover points along the array are randomly chosen and the in-between sections are exchanged. This process eliminates the single-point crossover bias towards the end segments of the array.

After a new solution is generated, we have to check whether all domain conditions are satisfied. If any of the constraints are not met, the solution is either discarded or modified to satisfy the conditions. For example, any newly-generated solution could be modified to guarantee that the opacity value corresponding to zero scalar data values is zero.

3.5 Solution Evaluation

The translation from encoded transfer functions to images is given by the direct volume rendering algorithm. This algorithm introduces a complex, nonlinear relationship between the transfer functions and the associated image. The most fundamental contribution of this paper is the idea of hiding this complexity from the user by transferring the problem from the evaluation of actual transfer functions (genotype) to the evaluation of the rendered images (phenotype). In other words, we simply hope and assume that a better transfer function produces a better image after rendering. "Better" in this context can be completely defined by the user's *subjective* intentions, or by some user-specified *objective* fitness function.

In the subjective approach, the user interactively evaluates the performance or fitness of transfer functions by analyzing the resulting images. This model allows the user to come up with a meaningful and satisfactory rendering without needing any knowledge on how the transfer functions look or how they influence the rendering process. Consequently, we change the basic stochastic search algorithm shown in Figure 2 to include a user-evaluation stage.

User evaluation of transfer functions is based purely on the quality of the images they generate. By selecting satisfactory images, the user selects the associated transfer functions as well. A fitness value f_i is assigned to each function according to the user selection. All functions have an initial fitness value of 0. Once they are selected, their fitness value becomes 1. We call such a scheme *binary selection* because it corresponds to a "like" or "don't like" decision of the user during evaluation of the images.

Other schemes are possible, for example, a subsequent reduction of the assigned fitness value in the order of selection. The function selected first gets a fitness of $f_1 = 1$. The fitness value f_i of the ith

selected function is reduced by a constant reduction value f_r, that is, $f_i = f_{i-1} - f_r$. However, we found that such a ranking scheme requires a fair amount of discipline and book-keeping from the user, and that binary selection is an easier and more natural way of user interaction.

In the alternative objective approach, we use automatic evaluation of images to generate renderings that satisfy certain *objective* criteria. The user first specifies one or more 2D image operators that measure a certain image quality. The optimization problem is to find images that minimize or maximize these measurements while satisfying domain conditions.

Several image operations have been incorporated into our system as the criteria of fitness evaluation. First, an approximation of the first order image entropy has been implemented to approximately measure the amount of information in the image. For example, images with high entropy generally contain a lot of details. Therefore, maximizing the image entropy can usually provide a good starting point for the exploration of unknown data. However, image entropy only measures the probabilities of gray values occurring in the image and is not concerned with what these values actually are. For example, an image consisting of 50% pixels with value 255 and 50% pixels with value 254 has the same entropy as an image consisting of 50% pixels with value 255 and 50% pixels with value 0.

An alternative measure is to maximize the variance of pixels in the final image. This typically leads to an image histogram that is spread out, although it has the danger of moving all the gray levels to the ends of the spectrum. The third function we incorporated is aimed at maximizing the edge energy in an image. The intention is to achieve images with well-defined surfaces, which are among the most interesting visualization results. The user has the choice of proportionally combining these criteria. Of course, a user can also use any other fitness evaluation function, depending on the application.

4 User Interface

To experiment with the proposed method for transfer function generation, we have tested a variety of stochastic techniques, which include hill climbing, parallel hill-climbing, load balance parallel hill-climbing, simulated annealing, and genetic algorithms. These techniques have been incorporated into VolVis [1], a public domain volume visualization system developed at SUNY Stony Brook. Figure 5 (included in the color section of these proceedings) shows the graphical user interface of our implementation.

The upper-left area of Figure 5 shows an image window with 25 volume-rendered images of a simulated high-potential iron protein dataset generated from a population of pre-defined initial transfer functions. Due to screen real-estate limitations, we chose to render only small, so-called "thumb-print" images of 100 by 100 pixels and restricted the population size S to 25. Note that the performance of stochastic search techniques, such as genetic algorithm, is closely related to the selection and the size of the initial population, and generally 25 is small for the population size. However, our experiments indicate fast convergence when user selection is used as the fitness evaluation. For objective evaluation, a larger population size can be used since intermediate images do not have to be displayed.

The images are rendered using the ray-casting routines of VolVis with a user-specified low to high accuracy. We currently use only opacity and color transfer functions. The rendering time is by orders of magnitudes longer than the execution time of the stochastic algorithms. To achieve interactive control, we calculate and maintain a 3D sample buffer which stores the values of the sample points along each ray at the beginning of the evolution. Thereafter, when a new set of transfer functions is applied to produce a new image, we need only to perform the compositing and thus save an enormous amount of time.

The upper-right portion of Figure 5 shows the control panel. In addition to image accuracy, color format (gray scale or RGB), evaluation method (manual or automatic), and stochastic technique, the user also has the flexibility of choosing the optical properties that need to be considered during optimization. Currently, we allow three choices: opacity and color, opacity only, or color only. Other values that may be changed include parameters for mutation splatting and low-pass filtering. Furthermore, the user can interactively modify the parameter values used in the evolution process. For example, the genetic algorithm contains three important parameters: the population size, the crossover rate, and the mutation rate. With our population size of 25, we chose an initial crossover rate R_c of 0.9 and a mutation rate R_m of 0.01, as suggested in [7]. The user can also interactively set the parameters for the fitness evaluation functions, such as the weights for entropy, variance, edge, and user-selected fitness functions. Note that user selection can be used as a part of the automatic fitness evaluation function, thereby combining the *subjective* and *objec-*

tive fitness evaluations. Furthermore, the system allows the user to define up to 10 domain conditions for each optical property.

The bottom area of Figure 5 shows the evolution process using a dynamic tree [13]. Each node of the tree corresponds to a generation of the evolution. At each node, a population of transfer functions is represented by an icon which shows the first image of the generation. The tree keeps record of the evolution history and thus allows the user to trace back to a particular generation.

When user evaluation is desired, an image of the parent generation can be selected with a single mouse click. Each selected image is highlighted and can be de-selected by a second mouse click. A double click on an image starts a pre-defined action, such as the generation of a higher resolution rendering of the image. Alternatively, a drawing window displays the color and opacity transfer functions associated with the image. The user can then interactively edit and change these transfer functions during the evolution process.

5 Results

Figure 6 (in the color section) shows the exploration of different volume visualizations of an MRI head. The resolution of the MRI dataset is $59 \times 133 \times 133$. The genetic algorithm and manual evaluations were employed to generate the images. This particular dataset is known to be quite difficult to visualize because slight changes in the transfer functions may produce dramatically different results. In this example, instead of converging to a set of similar images, we tried to get many different renderings using only one genetic evolution process. Consequently, we selected very different parents from a certain generation, with the hope that a new, interesting visualization would be generated after mutation and crossover. We also modified the genetic parameters and used high mutation and crossover rates. This example effectively presents the power of our algorithm for exploring unknown datasets or for generating new visualization effects.

Figures 7a and 7b (in the color section) show example evolutions towards different renderings of a high-potential iron protein with resolution of $66 \times 66 \times 66$ voxels. Each image has been rendered with high image accuracy using our optimized volume ray caster with the 3D sample buffer. It took about 10 seconds for both rendering and stochastic processing to generate a whole population of 25 images on a Silicon Graphics High Impact equipped with one 250MHZ R4400 processor and 128MB of RAM. Genetic algorithm and manual evaluation were employed for the evolutions.

Both figures show only the user-selected images, where each row corresponds to a different evolution generation. The top rows show user selections from the initial population, $P0$, which is shown in Figure 5. The boundary of each image indicates which optical property the user selected: red for color and opacity, green for opacity, and blue for color. The bottom row shows examples from the final population of images. Our intention was to rely on the genetic algorithm and manual evaluations to produce two very different sets of renderings from the same initial population. Figure 7a shows the evolution towards orange, mostly opaque images, whereas the final images of Figure 7b are green and mostly transparent. Please note the fast convergence of the algorithm towards the desired results after only two generations. No transfer functions were edited, and for each set to arrive at the final rendering took well below 5 minutes.

Figures 8a through 8d (in the color section) present results using *objective* fitness evaluation. Starting from the same initial population shown in Figure 5, we applied different stochastic algorithms for 100 generations. The evaluation criteria used are maximizing image entropy for Figure 8a, maximizing image histogram variance for Figure 8b, maximizing image energy content for Figure 8c, and maximizing the equal combination of these three criteria in Figure 8d. It can be seen that all the features that appear in any of the images of the initial population are present in Figure 8a, and that well-defined surfaces are visible in Figure 8c.

We have performed several experiments using different stochastic search techniques. Although we can not draw any substantial conclusions because of the limited number of experiments, it became clear that simple hill-climbing is not as effective as the other approaches. On the other hand, there seems to be no major difference between genetic algorithms and simpler stochastic approaches such as parallel hill-climbing and simulated annealing.

6 Conclusions and Future Work

In this paper we have presented a new approach for generating transfer functions for volume rendering using stochastic search techniques. The incorporation of global optimization techniques into a visualization system yields a novel type of user interaction with the dataset. Instead of dealing with the large and abstract parameter space of the rendering algorithm directly, the user can concentrate on the quality and meaning

of the resulting images. By interactively or automatically evaluating the performance of each image, the user guides the search process towards a set of final renderings satisfying visual or other goals. Our implementation demonstrates that optimal and sometimes unexpected but interesting results can be achieved without knowledge of the complex interaction between transfer functions and the rendering process.

We are currently implementing a number of improvements to the system. For example, our experiments show a strong relationship between the selection of the initial population and convergence speed. We think that it may be beneficial to incorporate methods of expert systems for the selection of the initial transfer functions. In addition, we plan to apply the proposed evolutionary methods to other fields in computer graphics, for example, to the evolution of spectral reflectivity functions for different surface materials, or to the generation of texture maps.

7 Acknowledgments

This work has been supported by the National Science Foundation under grant CCR-9205047 and by the Department of Energy under the PICS grant. We want to thank Tom Malzbender at HP Laboratories and Joe Marks at Mitsubishi Electric Research Laboratories for their helpful suggestions on the automatic fitness evaluation functions and stochastic search techniques. The high-potential iron protein dataset is courtesy of Scripps Clinic, La Jolla, CA; the MRI head dataset is courtesy of Siemens, Princeton, N.J.

References

[1] R. Avila, T. He, L. Hong, A. Kaufman, H. Pfister, C. Silva, L. Sobierajski, and S. Wang. VolVis: A diversified system for volume visualization research and development. In *Proceedings of Visualization '94*, pages 31–38, Washington, DC, October 1994.

[2] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette, editor, *Genetic Algorithms and Their Application*, Proceedings 2nd Intl. Conference, pages 14–21, Cambridge, MA, July 1987.

[3] J. F. Blinn. Models of light reflection for computer synthesized pictures. *Computer Graphics*, 11(2):182–198, July 1977.

[4] J. F. Blinn. Light refelction functions for simulation of clouds and dusty surfaces. *Computer Graphics*, 16(3):21–29, July 1982.

[5] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics*, 1(1):7–24, January 1982.

[6] K. A. DeJong. *Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Dept. of Computer and Communication Science, 1975.

[7] J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-16(1):122–128, January/February 1986.

[8] J. H. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[9] J. T. Kajiya and T. Von Herzen. Ray tracing volume densities. *Computer Graphics*, 18(3):165–173, July 1984.

[10] S. Kirkpatrick, C. D. Gerlatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, (220):671–680, 1983.

[11] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics & Applications*, 8(5):29–37, May 1988.

[12] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[13] S. Moen. Drawing dynamic trees. *IEEE Software*, pages 21–27, July 1990.

[14] B. T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, (18):311–317, June 1975.

[15] K. Sims. Evolving virtual creatures. In *Computer Graphics Proceedings*, pages 15–22. ACM Siggraph, July 1994.

[16] M. Srinivas and L. M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6):17–26, June 1994.

Figure 5: User interface with 25 initial images.

Figure 6: 25 different images for an MRI head.

Figure 7: Two transfer function evolutions with user evaluations for a simulated high-potential iron protein.

Figure 8: Final images generated by automatic evaluations of different criteria: (a) image entropy; (b) image variance; (c) edge content; (d) a combination (1/3 entropy, 1/3 variance, and 1/3 edge content).